FACEBOOK, INC.,

PLAINTIFF,

V.

POWER VENTURES, INC. DBA POWER.COM, ET AL,

DEFENDANTS

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

CASE NO. C-08-05780-JW

EXPERT REPORT OF BOB ZEIDMAN AND LAWRENCE MELLING

ZEIDMAN CONSULTING

DECEMBER 19, 2011

Table of Contents

We, Bob Zeidman and Lawrence Melling, on behalf of Zeidman Consulting, provide the following expert disclosures.

I. SUMMARY OF FINDINGS

1. Based upon the review of Defendants' source code for various code projects named PowerScript, PowerNavigtor, PowerProxy, and spider, as well as other documentation produced to date, we have concluded the following:

   (a) Defendants developed proprietary software named PowerScript and spider in order to crawl various social network websites, including particularly www.facebook.com ("Facebook"), to extract or "scrape" website user information such as Facebook photo images, wall content, friends' lists, and the like, and to then reformat that user information on Defendants' own website, www.power.com ("Power.com" or "the Power website"), in order to "proxy" Facebook and permit Defendants' own website users to log into Facebook through Defendants' own Graphical User Interface ("GUI"), rather than through Facebook's interface.

   (b) Defendants designed their proprietary PowerScript and spider software to automatically post on the Facebook website new Events soliciting Facebook users to join Power.com as part of what Defendants called the "Power 100" or "100x100x100" Campaign. Defendants likewise designed their software to automatically post Power Invitations on Facebook users' Walls soliciting them to join Power.com.

   (c) Based upon available information from Defendants' databases, at least 39,137 users of the Power website also had Facebook accounts. Because of missing information from those databases that is solely in the control of Defendants, we were unable to quantify exactly how many Facebook Event or wall posting transactions took place between the Power website and Facebook in which Facebook users were solicited to join Power.com. We are able to state that both kinds of solicitations did occur, however, and were initiated by Defendants' proprietary software.

(d) In addition to the electronic mail communications that Defendants' software automatically posted on the Facebook website when it created Facebook Events and when it posted Facebook wall messages, the same proprietary software that Defendants used to automatically create Event notifications and post Facebook Wall messages also would initiate automated "spam" email messages being sent on Defendants' behalf to Facebook users as a result of the software's ability to exploit Facebook's own email notification processes.

(e) Defendants designed their network architecture to circumvent technical barriers – such as blocks of IP addresses – that Facebook and other websites put in place to block the Power website's continued access. Defendants' source code includes routines that create a list of proxy servers. These proxy servers were continuously monitored by Defendants' software to determine if they were blocked by a website like Facebook. When blocked, the software could add a new host IP address for the PowerScript to access that would be employed to ensure continued access to the blocking website.

(f) Defendants formerly maintained "Power_Logger" and Async databases which logged information concerning the number of times Facebook users were contacted by the Power website and/or Power users. Among the information that was formerly contained in one or both of those databases was information identifying how many times Facebook users were sent invitations, either through Event notifications or Wall posts, to join Power.com as part of the "Power 100" or 100x100x100 campaign. That information, which was solely within the control of Defendants to document in its databases, has been deleted, preventing Facebook from knowing the true total number of spam invitations that were sent as a result of the execution by Defendants of their PowerScript software.

II. BACKGROUND

2. This introductory section of our report gives information about our qualifications.

A. PERSONAL EXPERIENCE AND BACKGROUND OF BOB ZEIDMAN

3. Robert is an engineer and the founder and president of Zeidman Consulting, which provides engineering consulting services to high-tech companies. Among the types of services Robert provide are hardware and software design. My clients have included Fortune 500 computer and technology companies as well as smaller companies and startups. A copy of my resume is attached hereto as Exhibit A.
4. Robert holds a Master's degree from Stanford University in Electrical Engineering and two Bachelor's degrees from Cornell University, one in Electrical Engineering and one in Physics.
5. Robert has been a computer software and hardware designer for over 25 years. Robert have designed and developed a variety of computer hardware and software products. These software products include Internet-based training courses and web-based course administration software, an operating system synthesis tool, a source code comparison tool, a network emulation software bridge, and a remote backup system whereby user data is automatically transmitted and stored at a remote location. Robert have founded several companies including eVault, a remote backup company; the Chalkboard Network, an e-learning company; Zeidman Technologies, a company that develops software tools for enabling and improving hardware and software development; and Software Analysis and Forensic Engineering Corporation, a company that develops software analysis tools.
6. Robert has written a variety of papers, books, and presentations on computer hardware and software and other engineering subjects. Robert am the developer of the Universal Design Methodology, a process for efficiently developing reliable systems, about which Robert have written extensively. A list of my publications is included in my resume attached as Exhibit A.
7. Robert holds a number of patents for software synthesis, hardware emulation, hardware synthesis, hardware simulation, and software code comparison. Robert have created a tool called CodeSuite® that incorporates BitMatch®, CodeCross®, CodeDiff®, CodeMatch®,

CodeCLOC®, and SourceDetective® for detecting whether one computer program has been plagiarized from another computer program.

8. Robert has consulted on matters involving intellectual property disputes, including instances of alleged misappropriation and infringement. My work in this capacity has included, among other things, reviewing and analyzing software source code, reviewing and analyzing patents, reverse engineering hardware and software, writing expert reports, and testifying in court.
9. Robert has testified at deposition and at trial in a number of cases involving software copyright infringement, trade secret theft, and patent infringement. The specific cases can be found in my resume, attached as Exhibit A.

B. PERSONAL EXPERIENCE AND BACKGROUND OF LAWRENCE MELLING

10. Lawrence is a research engineer at Zeidman Consulting. Lawrence has over 30 years of executive management and engineering experience in developing new hardware and software technologies and bringing them to market. Lawrence has been engaged in applications engineering and marketing of electronic design automation (EDA) tools at major companies and small startups. Lawrence has also been involved in the development of sophisticated tools for source code and object code analysis for finding intellectual property infringement. Lawrence has not previously testified at trial or in a deposition. My resume is attached as Exhibit B to this report.

III. DEFINITIONS

This section provides a discussion of technical terms needed to understand this report, which we are prepared to further explain at trial.

A. WEBSITE

11. A "website" is a location on the World Wide Web that contains a group of web pages typically created using a popular programming language called the Hypertext Markup Language (HTML). Websites are usually connected to each other using "hyperlinks," and

are made available to the public by an individual, company, educational institution, government body, or other organization. These web pages are hosted on one or more computers called "web servers" and are viewed by users on "client computers" that are connected to the web servers via the Internet. The web pages are viewed using an Internet browser, such as Microsoft's Internet Explorer. In conjunction with this Expert Report, we make extensive reference to two websites located at the Uniform Resource Locators ("URLs") http://www.facebook.com (the "Facebook website") and http://www.power.com ("Power.com" or the "Power website").

B. INTERNET BROWSER

12. An "Internet browser" or web browser is a typical client application used to navigate the Internet. The browser accesses information such as web pages, images, videos, and games from Internet servers. The URL is the "address" through which online information is located and retrieved by the user from her client computer. Servers may provide static information to an Internet browser or may dynamically generate the information that is transmitted to an Internet browser based on input from the user and the internal state of the server. The browser provides the graphical user interface (GUI) to the web pages on the server. However, some websites make use of client-side software to offload processing from the server to use the client's computer. This is important because the browsers include functionality to execute client-side "web scripts," which concept we discuss below. Three popular Internet browsers in use today are: Microsoft's Internet Explorer, Mozilla's Firefox, and Google's Chrome.

C. CLIENT

13. A "client" is a computer that makes a service request to a server (defined below); the server fulfills the request. Computer interactions using the client/server model are very common. For example, when an individual checks a bank account from his or her computer, a client

program in the individual’s computer forwards the request to a server program at the bank. The bank’s program may respond, or it may, in turn, forward the request to its own client program that makes a request to another bank computer. With regard to the World Wide Web, the browser on an individual’s computer is a client program. A client application can also be referred to as the “front-end” and the server application is often called the “back-end.”

D. SERVER

14. A “server” is a computer on a network (such as an internal corporate network or the Internet) that is dedicated to a particular purpose; it stores information and performs critical functions. For example, a “database server” could store all of an organization’s data on a single machine, while providing database services to multiple users anywhere in the office, or even the world, and while also allowing access and control over the data. A typical “database server” will allow users to utilize their data from custom applications designed to meet their specific needs. Server software refers to software running on the server computer that “serves up” information to a client computer. With regard to the World Wide Web, a web server responds to web client requests to view web pages. These pages can be static (content doesn’t change) or dynamic (content is determined when requested).

E. PROXY SERVER

15. A proxy server is a machine used to relay Internet transactions between clients and websites such that the transactions with the website appear to originate from the proxy server's IP address. An IP address is the Internet Protocol address used to identify a machine, such as a server or proxy server, connected to the Internet. Proxy servers are used for a number of purposes, including the following activities: (a) keeping machines behind the proxy (such as the website’s actual host servers) anonymous; (b) speeding up access to resources frequently used by multiple users behind the proxy by using caching techniques; (c) controlling access

to website content or services; (d) accessing websites from a computer whose own IP address otherwise would be blocked by the accessed website; (e) logging or auditing Internet use; and (f) circumventing security procedures or controls aimed at limiting access to or blocking a particular IP address.

F. WEB SCRIPTS

16. "Web scripts" are written to generate dynamic web pages -- that is, web pages with rapidly changing content and imagery or content that must be somehow calculated via software mechanisms. For example, webscripts can be used to calculate and display the total visitor count to a website. Such scripts are written in a variety of scripting languages such as PHP, CGI, Perl, and JavaScript. Some scripts run on the web server (server-side), while other scripts run on the user's machine (client-side). Such webscripts also can be embedded within HTML in order to affect the behavior of web pages. Of the languages mentioned, JavaScript is the language of choice for client-side scripting and is supported by all the Internet browsers popularly in use, while PHP, CGI, and Perl are popular for server-side scripting.
17. In Microsoft Windows systems, component-based scripting is implemented through a technology called "Active Scripting," and employs what are commonly called "script engines." One particularly popular form of a server-side Active Scripting engine is called ASP, or "Active Server Pages," which is used to develop dynamically-generated web page content.

G. WEB CRAWLER OR SPIDER

18. A "web crawler" or "spider" is a computer program used to browse the Internet in a systematic, comprehensive way. Web crawlers are typically associated with search engines and are used to collect website information for search engine indexing. Nonetheless, spiders and web crawlers are now commonly being used to collect or "harvest" web page information for non-search related applications such as web scraping. Web scraping can be

used to locate input fields and variable fields that allow a program to automatically fill out forms to login, send messages, request information, or any other website activity initiated by the filling out of a form. Because web scraping often is employed by entities for unwanted or unlawful purposes (like Defendants' harvesting of user information such as "friends' lists," and similar data from Facebook in order to later use that information to send "spam" email and electronic mail messages), many website operators (including Facebook) publish Terms of Use provisions that prohibit the use of web scrapers on their websites by their registered users.

H. COMPUTER DATABASE

19. Computer databases consist not only of data, such as user names and addresses, but also consist of schema and procedures represented by source code. The term "schema" refers to the structure of the database, including where to place the data, how to organize the data, and the particular relationships between the data. For example, customer names may be placed in a field called "Name," and that name is referenced in a table called "Customers." A table can be visualized as a spreadsheet and the field would correspond to a particular column in the spreadsheet. In a database there are many different tables. Each customer name may have an associated table that has fields that contain the customer's address, credit card number, account balance, and comments about the customer. The table names, field names, types of data in the fields, and relationships between different tables and different fields constitute the schema of the database, which is described using a special programming language such as the Structured Query Language, also known as SQL. Two popular forms of SQL servers are MySQL, an open source relational database management system, and Microsoft SQL Server (MSSQL).

20. Procedures for manipulating the data may also be stored in databases and are represented by a special programming language such as SQL. These "stored procedures" can be used by programs that access the database to manipulate the data in the database. For example, a

stored procedure may exist to compute the average outstanding balance for a list of customers. A program that is written to access the database could also access the stored procedure in order to calculate this average.

I. SQL SERVER

21. Microsoft SQL Server (MSSQL) is a relational database management system. A relational database is a sophisticated method of grouping data together based upon common attributes to provide greater speed and reliability for data access.
22. SQL tables will often involve common English words, but the sequence of items is usually arbitrary. The tables are used to organize data, and do not have to be in any specific order to function correctly. Each specific category of data is known as a tuple. There is no order imposed upon how the tuples are organized. The order of tuples in a relational database is arbitrary. If similar or identical sequences of elements are found in a SQL table, it can be a sign of copying despite the elements having common names.

J. SOURCE CODE

23. In computer science, "source code" is a kind of text that is written using the format and syntax of the programming language that it is being written in, and typically is the only format that is readable by humans. Computer programs can be written using complex instructions that look like English. For example, the instruction a = b*c+2 tells the computer to take the number stored in memory and represented by variable b, multiply that by the number stored in memory and represented by the variable c, add 2 and store the result in memory represented by the variable a. Similarly, the statement printf("Hello world!") tells the computer to print the words "Hello world!" to the computer screen. These high-level, English-like instructions are the "source code." Computer programs are made up of many lines of source code and the process of writing these lines of code is called programming. Eventually these lines of source code are turned into instructions that a computer

understands, consisting of sequences of electronic ones and zeroes. The process of turning human-readable source code into a file containing computer instructions is called "compiling" and is performed by a special computer program called a "compiler." In some cases, source code is run directly by a computer, without creating any file of computer instructions.

24. Source code comes in a variety of programming languages, some of which are called "low level" programming languages, and others which are called "high level" programming languages. PHP, Perl, Java, JavaScript, and SQL all are examples of high level programming languages. Other popular examples of high level source code programming languages are ones called C, C++, C#, Smalltalk, APL, AppleScript, Ruby and Python.

IV. SCOPE OF REPORT

25. Based on our background and experience, we have been asked to provide our opinions and conclusions related to (1) whether Defendants' source code contained evidence of attempts by Defendants to access Facebook, scrape Facebook, download data from Facebook, contact Facebook, and/or use information scraped/and or downloaded from Facebook to "proxy" the Facebook website; (2) whether Defendants' source code reflects evidence that Defendants used their software to establish Facebook Events and/or wall messages inviting Facebook witnesses to automatically receive electronic mail messages or email messages inviting them to join the Power website; (3) whether Defendants developed technology to circumvent any attempted block by Facebook of the IP addresses used by Power.com to connect with Facebook; and (4) whether there is evidence that Defendants ever included information related to their Power100 (or 100x100x100) marketing campaign in their Power_Logger or Async databases. We have reviewed literally hundreds of thousands of lines of code to reach our opinions. In addition, in reaching the opinions and conclusions discussed herein, we received, considered, and/or relied upon the following materials, copies of which are not attached but can be provided upon request:

- Power Source Code Documents, which now include 5,743,505 lines of code.
- Sixty-nine SQL Server database backup files.
- We used Understand by Scientific Toolworks, Inc. to help analyze the software.
- Microsoft SQL Server 2008 to extract the databases from the backup files and to review the database contents.
- Fifty-five PowerScript Source files extracted from the PowerScript_bkp_full.bak database backup file.
- Numerous source code files provided as exhibits to this report.
- The transcript and Exhibits from the July 20, 2011 deposition of Defendant Steve Vachani, and the testimony from, and Exhibits used at, the December 14, 2011 deposition of Zak Mandhro.
- The December 12, 2011, Declaration of Steve Vachani in Support of Defendants' Oppositions to Facebook's motions for summary judgment.
- Facebook's source code for "Create an Event."
- Emails, technical documents and marketing documents produced by Defendants and third-party witnesses in discovery in this litigation, which are referenced in this Report.

26. We have been retained to review and analyze the source code and databases produced by Defendants in this action. We reviewed code and databases produced by Defendants on August 25-26, 29-30, September 6-7, October 19 and 25, 2011, November 1-4, 7-9, 11, 16, and 19-21, December 12-13 and December 15-16. Copies of our "Power Source Code Inspection Logs" maintained in accordance with the Protective Order entered in this case are attached hereto and combined as Exhibit C.

V. COMPENSATION

For the work of Lawrence Melling on this matter Zeidman Consulting is being compensated at a rate of $200 per hour. For the work of Bob Zeidman, Zeidman Consulting is being compensated at a rate of $750 per hour

VI. ANALYSIS

Our analysis is broken into five sections:

- We analyze how Defendants' software was used to connect to the Facebook website, spider the Facebook website, scrape Facebook user content and user information from the Facebook website, download Facebook user information and user content to the Power website, and to emulate or "proxy" Facebook as part of the Power website's social aggregation services.
- We analyze how Defendants' software was used to initiate spam emails via the PowerScripts developed to create content on Facebook. We provide a detailed analysis of the CREATE_EVENT_FACEBOOK and PN_SEND_SCRAP_FACEBOOK scripts that were used to create Facebook Events for the Power 100x100x100 campaign, and which were also used to post Power invitations on Facebook users' Walls.
- We discuss how the Power databases identify Facebook information stored in the databases. We also show why we were unable to determine how many Power.com transactions occurred with Facebook users, because the relevant information was deleted some time after it was originally stored.
- We analyze Defendants' efforts to circumvent Facebook's IP Blocks: The Power proxy system developed to manage and control a pool of proxy servers used to access sites like Facebook in order to circumvent IP blocks like the ones put in place by Facebook.

- We have also included a short section providing a review of the technical accuracy of certain arguments made by Defendant Steve Vachani in his December 12, 2001 declaration.

A. DEFENDANT'S SOFTWARE USED TO CONNECT TO THE FACEBOOK WEBSITE, SPIDER THE FACEBOOK WEBSITE, SCRAPE FACEBOOK USER INFORMATION FROM THE FACEBOOK WEBSITE, DOWNLOAD FACEBOOK USER INFORMATION TO THE POWER WEBSITE, AND TO "PROXY" FACEBOOK

27. In analyzing the Defendants' source code, we determined that there were two core software components developed to retrieve information and post information to social network sites like Facebook. The two components are the PowerScript system and the PowerProxy system. The PowerScript system is best described as a web scraping system. A web scraper is software that can programmatically access web sites and perform operations intended to be done by a person, such as filling out forms, sending messages, and reading content. Web scrapers are also referred to as webbots, or simply "bots." Web scraping is generally not allowed under most websites' terms of use, and is considered a form of pirating by those websites. Also, because programmed transactions with a website can occur much faster than human transactions, a website's access can be slowed down or halted through a rapid succession of programmed transactions. Web sites that do allow other sites to programmatically access information would typically offer these services through a web service interface or an API (application program interface), as Facebook does with its Facebook Connect service, in order to manage and control these programmed transactions and maintain a reliable website.

28. Because web scraping is prohibited by most websites, one challenge to creating a web scraper is to avoid detection. One of the easiest ways to detect a web scraper is to look at the number of transactions coming from a specific Internet Protocol Address (IP address). To avoid detection it is common for sophisticated web scrapers to use proxy servers to scrape information. A proxy server is a machine that acts as a relay, so the website sees the IP address of the proxy server, and not of the actual machine running the scraper. By using a

pool of proxy servers it is possible to reduce a website's ability to detect the scraper by dispersing the transactions across the pool of proxy servers. Additionally, if one of the proxy servers is detected, the other servers can continue to maintain access even if the website blocked access for the detected server.

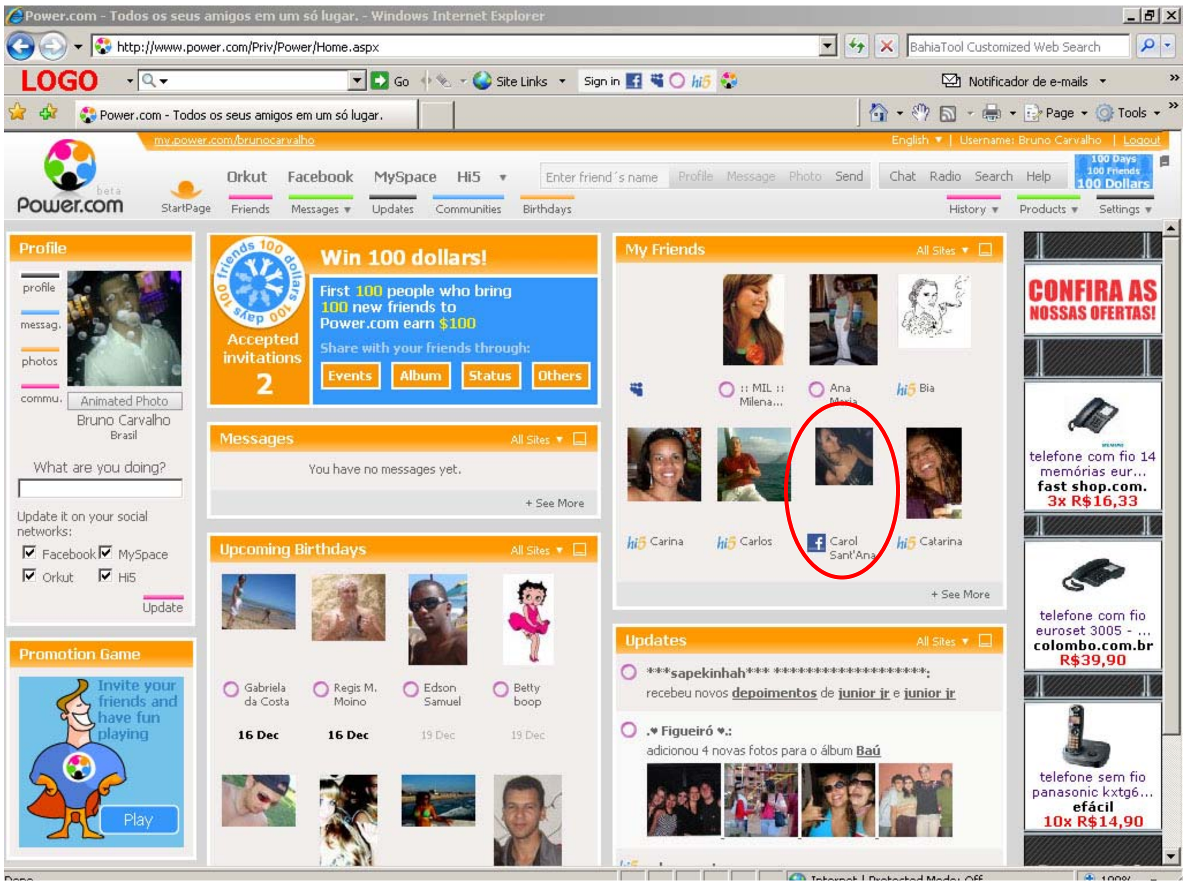
29. Our analysis shows that the Defendants developed the PowerScript system and PowerProxy system to scrape information from Facebook, proxy the Facebook website and avoid detection when engaged in such activities. In addition, our analysis shows how Defendants' programmed access initiated actions that resulted in unwanted commercial "spam" messages being sent to Facebook users soliciting them to join Power.com.

30. We analyzed 33 out of 55 PowerScripts written to perform transactions with Facebook's website. PowerScript is a scripting language developed by Power to programmatically obtain information from web pages, and write or post information, to web pages without requiring user interaction. Table 1 categorizes the scripts we reviewed. Of those, scripts use HTTP GET to read information from the Facebook website, and others use HTTP POST to post information on the Facebook website. These scripts were developed by Defendants for performing transactions on the Facebook site, are specific to Facebook, and would not work if targeted to another site. The developers writing these scripts would have been required to access Facebook via a Facebook user account to examine the HTML source in order to write a script to get or post Facebook information.

PowerScript Name	**Function**
PN_LOGIN_FACEBOOK	Post to login to Facebook
PN_VALID_CONTEXT_FACEBOOK	Get logout link to verify login is active
accept_friend_invitation_FACEBOOK	Post to accept Facebook friend invitation
CREATE_EVENT_FACEBOOK	Post to create Facebook Event and invite list of friends or all friends if no list is provided
JOIN_COMMUNITY_FACEBOOK	Post to join a Facebook group
PN_SEND_SCRAP_FACEBOOK	Post message to Facebook friend Wall
PHOTO_CREATE_ALBUM_FACEBOOK	Post a new Facebook photo album
PN_SEND_PRIVATE_MESSAGE_FACEBOOK	Post send private Facebook message to a Facebook friend
TUBESPREE.PutQuickEmbedInFacebook	Post new video link to Facebook

PN_SET_STATUS_FACEBOOK	Posts Facebook status update
PN_GET_FRIEND_PICKER_FACEBOOK	Get Facebook friend id and name
PN_GET_BIRTHDAYS_FACEBOOK	Get Facebook friends' birthdays
PN_GET_COMMUNITIES_FACEBOOK	Get Facebook group information (id, name, description, photo link, number of members)
PN_GET_PRIVATE_MESSAGE_FACEBOOK	Get Facebook messages for subject, message, friend Id, name, photo and message link, reply link, PrivateLock
PN_GET_ALBUM_LIST_FACEBOOK	Get Facebook photos for album id, name, date, dateorder, image, link
PN_GET_ALL_SCRAP_MESSAGE_FACEBOOK	Get Facebook Wall posts and messages - messages same as PN_GET_PRIVATE_MESSAGE_FACEBOOK and wall posts the same as PN_GET_SCRAP_FACEBOOK
PN_GET_FRIENDS_INVITATIONS_FACEBOOK	Get friend invitations for friend id, photo, name, and link
PN_GET_FRIENDSUPDATES_FACEBOOK	Get friend update, name, id,fullname, and update link
PHOTO_GET_ALBUM_LIST_FACEBOOK	Get list of photo albums
GET_HTML_PAGE	Get an entire HTML page
GETALBUMLIST_FACEBOOK	Get photo album list
OBTERIMAGEMFACEBOOK	Get photo page
GET_PROFILE_FACEBOOK	Get profile information (id, name,photo link, gender,birthday, email, phone, mobile phone,website link, city, country, relationship status, interests, favorite music, favorite TV shows)
PN_GET_SCRAP_FACEBOOK	Get Wall for friends photo, ID, Name, Date of post, content, encoded content, post id, post link, subject and showPrivateLock
PN_GET_AMOUNT_COMMUNITIES_FACEBOOK	Get/counts number of groups
DELETEALBUMPHOTO_FACEBOOK	Post to remove a Facebook photo album
PN_DELETE_SELECTED_PRIVATE_MESSAGE_FACEBOOK	Post to remove Facebook message
UNJOIN_COMMUNITY_FACEBOOK	Post to remove user from Facebook group
PN_DELETE_SELECTED_SCRAP_FACEBOOK	Post to remove Wall message
PHOTO_DELETE_PHOTO_FACEBOOK	Post to remove a Facebook photo
GET_VIDEO_FACEBOOK	Get a Facebook video (name, id, url, thumbnail, width, height)
PN_GET_FRIENDS_FACEBOOK	Get a list of friends(ids, names,and photo urls)
PN_LOAD_ATTRIBUTES_FACEBOOK	Gets name, id, photo url, gender, country

Table 1: PowerScripts analysis summary


Power.com - Todos os seus amigos em um só lugar. - Windows Internet Explorer

http://www.power.com/Priv/Power/Home.aspx

my.power.com/brunocarvalho

English | Username: Bruno Carvalho | Logout

Profile

Bruno Carvalho
Brasil

What are you doing?

Update it on your social networks:
☑ Facebook ☑ MySpace
☑ Orkut ☑ Hi5

Update

Win 100 dollars!

Accepted invitations 2

First 100 people who bring 100 new friends to Power.com earn $100

Share with your friends through:

Events Album Status Others

Messages

You have no messages yet.

+ See More

Upcoming Birthdays

Gabriela da Costa — 16 Dec
Regis M. Moino — 16 Dec
Edson Samuel — 19 Dec
Betty boop — 19 Dec

My Friends

:: MIL :: Milena...
Ana Maria
Bia
Carina
Carlos
Carol Sant'Ana
Catarina

+ See More

Updates

sapekinhah *******************:
recebeu novos depoimentos de junior jr e junior jr

.♥ Figueiró ♥.:
adicionou 4 novas fotos para o álbum Baú

Promotion Game

Invite your friends and have fun playing

Play

CONFIRA AS NOSSAS OFERTAS!

telefone com fio 14 memórias eur...
fast shop.com.
3x R$16,33

telefone com fio euroset 3005 - ...
colombo.com.br
R$39,90

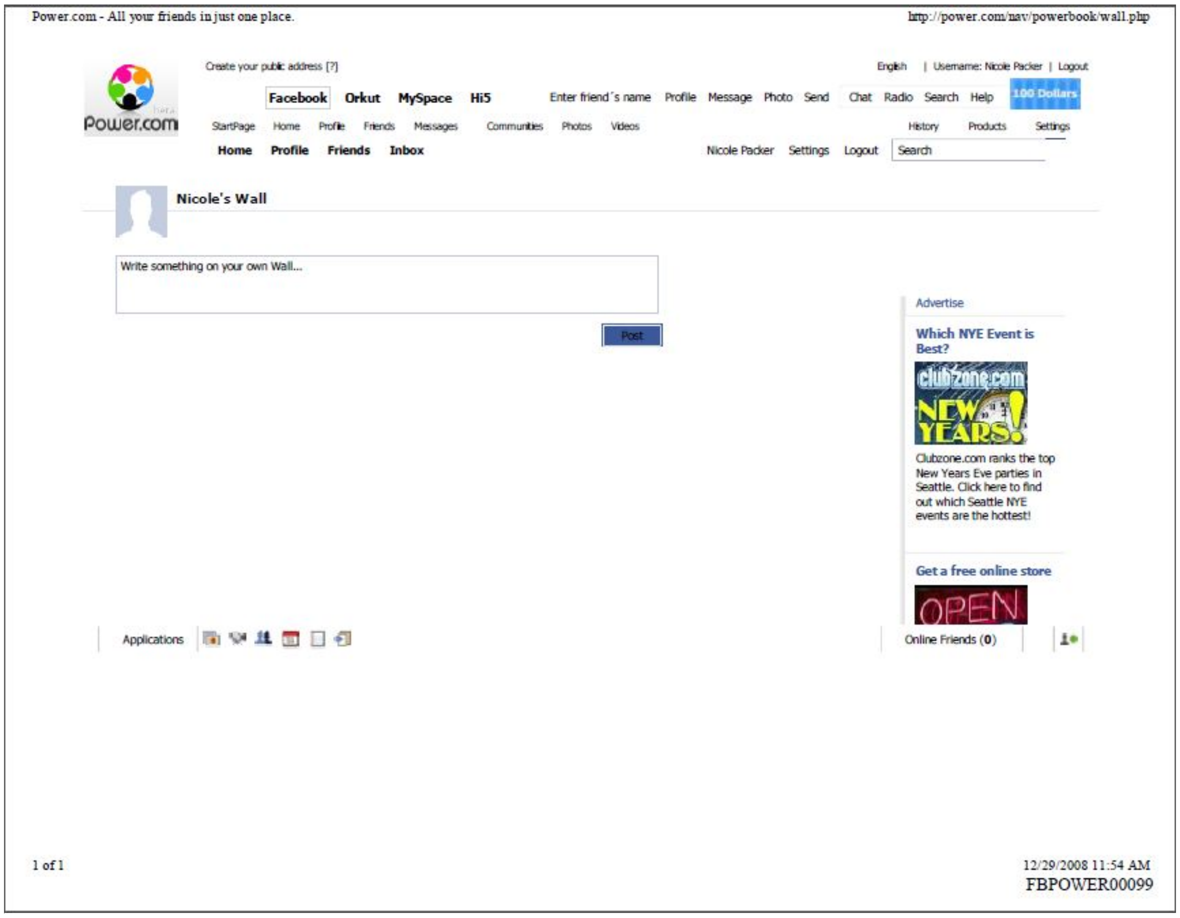
telefone sem fio panasonic kxtg6...
éfácil
10x R$14,90

Figure 1: Power.com Screenshot including a Facebook friend

31. The Power.com screenshot (found on page 2 of the toolbar_en.ppt presentation in directory SVN\apresentacoes\20081210 – toolbar) shown in Figure 1 is an example of how information from Facebook, gathered using PowerScripts, including the ones analyzed in Table 1, was included and framed inside the Power.com web page. In this example the scraped information includes a Facebook friend's photo (see red circle).

32. Defendants sometimes misleadingly call their Power.com website a "browser." For instance, Mr. Vachani referred to Power.com as a browser both at his deposition and in his December 12, 2011 Declaration. We don't believe that is an accurate description of Power.com's functionality. From our examination of Defendants' source code, and as further discussed below in conjunction with our discussion of Mr. Vachani's December 12, 2011 Declaration, Defendants developed software called Power Navigator which supported a browser style interface where the Facebook website could be displayed within Power.com, as shown in

Figure 2.


Power.com - All your friends in just one place.
http://power.com/nav/powerbook/wall.php
Power.com
Create your public address [?]
English | Username: Nicole Packer | Logout
Facebook Orkut MySpace Hi5
Enter friend's name Profile Message Photo Send
Chat Radio Search Help
100 Dollars
StartPage Home Profile Friends Messages Communities Photos Videos
History Products Settings
Home Profile Friends Inbox
Nicole Packer Settings Logout Search
Nicole's Wall
Write something on your own Wall...
Post
Advertise
Which NYE Event is Best?
clubzone.com NEW YEARS!
Get a free online store
OPEN
Applications
Online Friends (0)
1 of 1
12/29/2008 11:54 AM
FBPOWER00099

Clubzone.com ranks the top New Years Eve parties in Seattle. Click here to find out which Seattle NYE events are the hottest!

Figure 2: Screenshot of Facebook webpage embedded in Power.com web page

However, as we noted, the majority of PowerScript scripts directed to Facebook were not directed to browsing functions, but instead were written to programmatically obtain and post information to Facebook without user interaction. Such functionality cannot rationally be called browsing..

B. DEFENDANTS' SOFTWARE USED TO INITIATE SPAM EMAILS

33. We also examined two of the PowerScripts (`CREATE_EVENT_FACEBOOK` and `PN_SEND_SCRAP_FACEBOOK`) in more detail and found that each was responsible for initiating a sequence of programmed transactions to create a Facebook Event and post

Facebook Wall messages. These PowerScripts require Power's software and infrastructure to execute and, once executed, resulted in SPAM electronic messages being sent to Facebook users. These scripts were developed for a specific purpose, and that was to automate the creation of Facebook Events and posting of Facebook Wall messages.

34. The `CREATE_EVENT_FACEBOOK` script automatically set Power as the host of the event, and identified Power as the "location" for the event in Facebook's Event tool (see Exhibit D, `CREATE_EVENT_FACEBOOK.xml`, at lines 37 and 41).

35. The script also generated a guest list if one was not provided. To generate the guest list, Defendants' software accesses the user's Facebook "friendsList" and extracts the user ID of each friend to create the guest list. See Exhibit D, at lines 46-51. The PowerScript executes this code, if no guest list is provided, to automatically create a guest list from the user's list of friends on Facebook. Specifically, the PowerScript application checks a "variable" (a named element to store information) called the Guestlist ("`listaConvidados`"), and then executes a sequence of programming commands inside a "rule block," identified by the beginning tag "`<rule>`" and terminated by the ending tag "`</rule>`," if it is empty. Through this process, the PowerScript software creates a new variable called "`friendsList`," and another variable called "`ids`," which combine to create the Event guest list made from one Facebook user's list of Facebook "friends."

36. The script also automatically sends Facebook Event invitations to each Facebook user in the guest list on behalf of the Power website (see Exhibit D, at lines 58-74), and these Event invitations initiate spam messages being sent to the Facebook invitees.

37. We also looked to determine if Defendants, or the user, caused the Facebook "Events" to be initiated. From the code that has been provided to date, we could not locate any code in `CREATE_EVENT_FACEBOOK` that requested the user's approval to send the "Event" invitations. We also were unable to find any other code requesting that the user accept or approve sending the Facebook Event invitations on behalf of the Power.com website.

38. The PowerScript software also created the text used to invite Facebook friends to

participate in the “100x100x100” campaign. The message contents were stored in resource files, which are files used by Microsoft Visual Studio development tools to store information for access by a program. Notably, in this example there were three resource files found with the same content in three different languages: English, Spanish, and Portuguese (*see* Exhibit E, `PowerCallBack.aspx.en.resx`, found in directory `SVN\power.com\Power.Com\Pub\Http\App_LocalResources`, at lines 132-137).

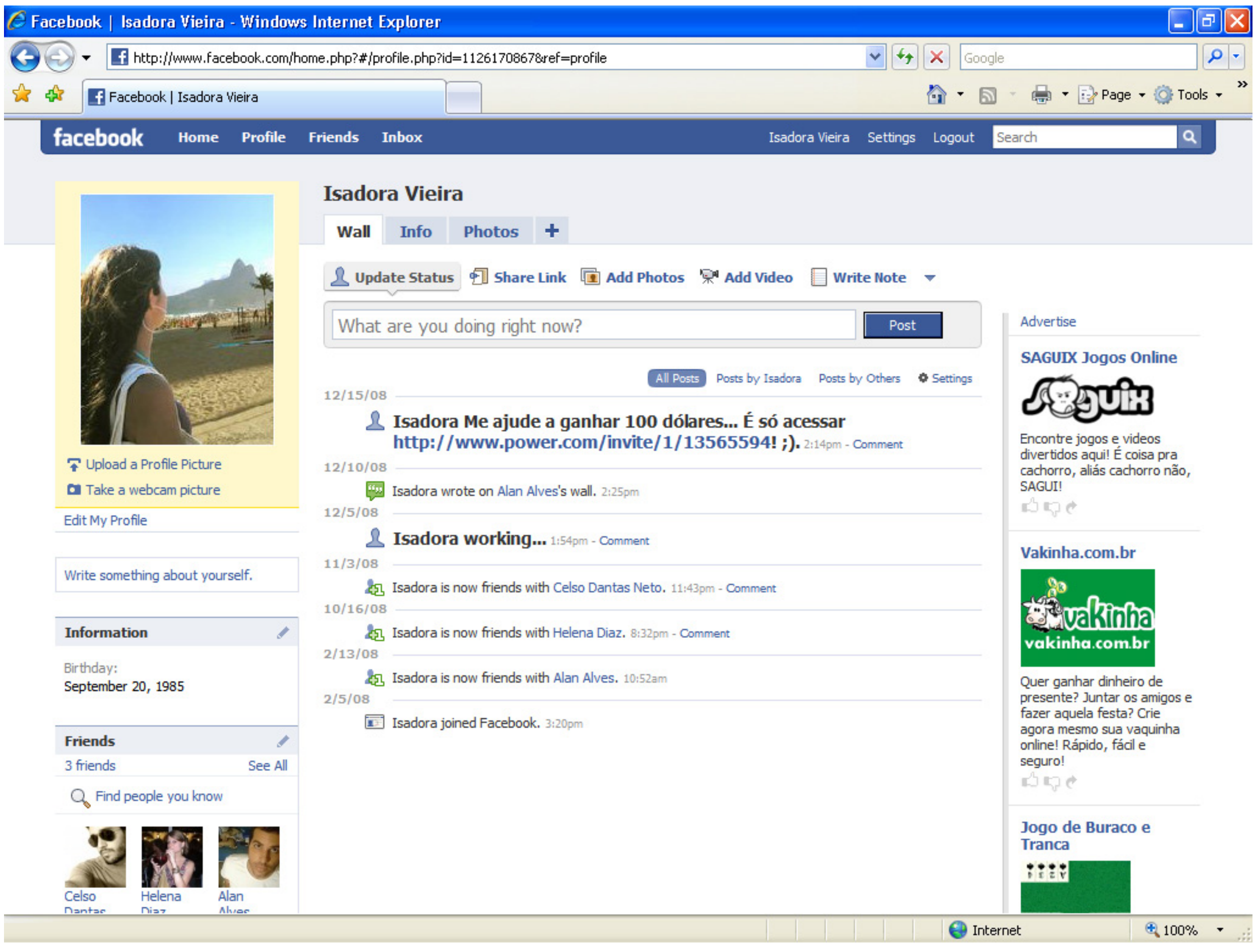
```
132 <data name="CAMPAIGNMESSAGE" xml:space="preserve">
133  <value>#BREAK##BREAK#I am competing for the $100
prize in the 100x100x100 promotion and recommend you to
participate too!#BREAK#Learn more at:</value>
134  </data>
135  <data name="CAMPAIGNMESSAGE2" xml:space="preserve">
136  <value>First 100 people who bring 100 new friends to
Power.com earn $100. Come and participate too:</value>
137  </data>
```

These messages were created and authored by Defendants to promote joining Power.com. When used with the `CREATE_EVENT_FACEBOOK` script, the messages would result in all of a Power.com user’s Facebook friends being automatically invited to join Power.com, and the friends then receiving spam emails as a result.

39. These strings include the actual language that was sent to Facebook users as a result of the Power.com website’s execution of the “`CREATE_EVENT_FACEBOOK`” script. The excerpt above shows that the text string stored for `CAMPAIGNMESSAGE` and another for `CAMPAIGNMESSAGE2` are both human-readable messages used in promoting the 100x100x100 campaign to Facebook users.

40. The html code for the Power.com web page that would initiate the creation of Facebook Events for this campaign was not found in the software sources provided. We believe this omission arises from the fact the Power 100 campaign was from an earlier date than the source provided. Since the source repository that would allow us to return to earlier software

releases was corrupted when we received it from Defendants, we were unable to find the html code that initiated these campaign events. However, we know from other sources, such as Mr. Vachani's deposition and the PowerScript source code that we reviewed, that such initiation of Facebook Events by the Power.com web page did, in fact, occur. For instance, in Figure 3, we offer a screenshot that shows a Power 100x100x100 campaign message posted on a Facebook user's Wall. This screen capture image produced by Defendants corroborates the occurrence of the Facebook Event and Wall posting transactions. See facebook.jpg found in directory `SVN\apresentacoes\20090120 - Intersite Connect\Source\ícones` image file:



Isadora Vieira

What are you doing right now?

12/15/08
Isadora Me ajude a ganhar 100 dólares... É só acessar http://www.power.com/invite/1/13565594! ;). 2:14pm - Comment

12/10/08
Isadora wrote on Alan Alves's wall. 2:25pm

12/5/08
Isadora working... 1:54pm - Comment

11/3/08
Isadora is now friends with Celso Dantas Neto. 11:43pm - Comment

10/16/08
Isadora is now friends with Helena Diaz. 8:32pm - Comment

2/13/08
Isadora is now friends with Alan Alves. 10:52am

2/5/08
Isadora joined Facebook. 3:20pm

Write something about yourself.

Information
Birthday:
September 20, 1985

Friends
3 friends See All
Find people you know
Celso Dantas
Helena Diaz
Alan Alves

SAGUIX Jogos Online
Encontre jogos e videos divertidos aqui! É coisa pra cachorro, aliás cachorro não, SAGUI!

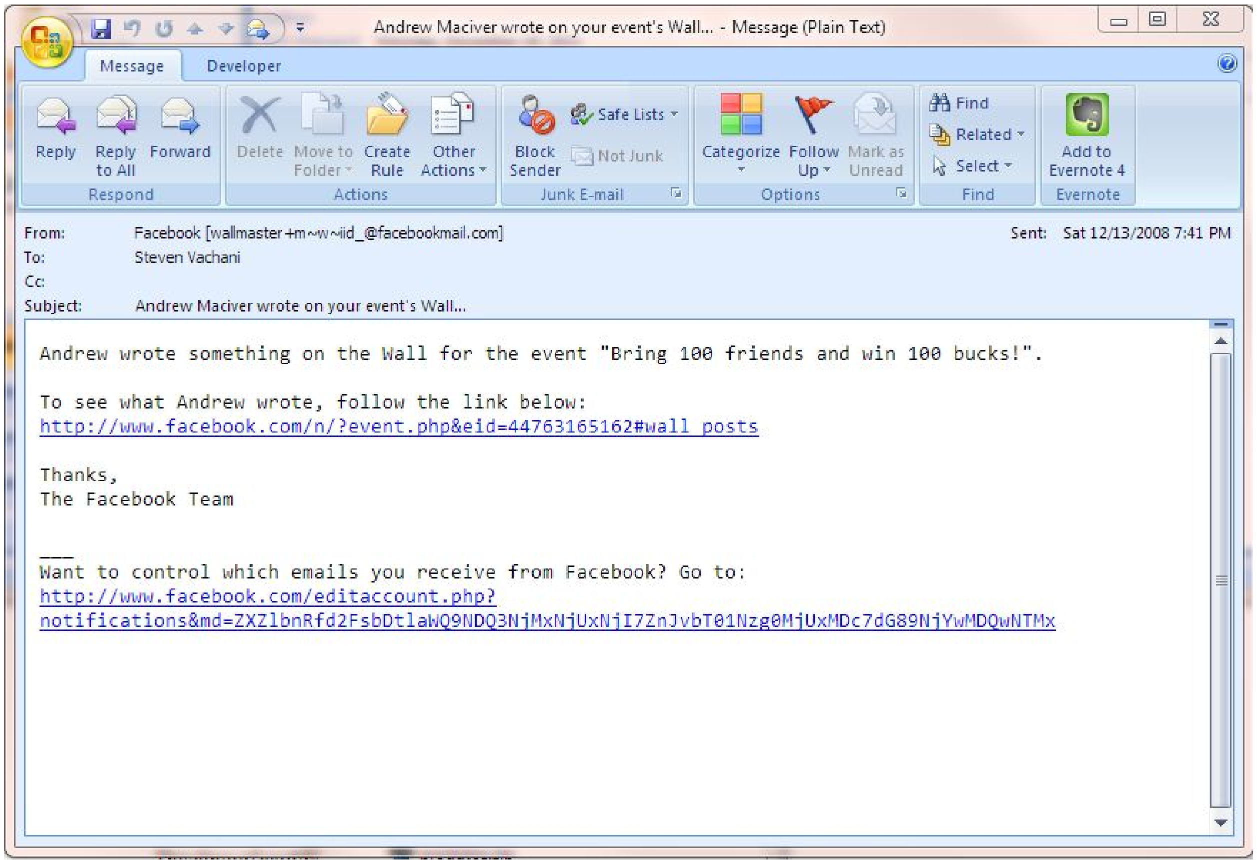
Vakinha.com.br
Quer ganhar dinheiro de presente? Juntar os amigos e fazer aquela festa? Crie agora mesmo sua vaquinha online! Rápido, fácil e seguro!

Jogo de Buraco e Tranca

Figure 3: Facebook Wall screenshot showing Power 100x100x100 campaign invitation

41. Additionally, we were able to examine some of the actual email messages sent when a wall

message was posted in response to Defendants creating an Event to invite a user to the Power100 or 100x100x100 campaign. The messages shown in Figure 4 are actual emails sent to Defendant Steve Vachani when his friends used the Power.com site to execute the PowerScript software made available through the html code to create Power 100 Events, and to thereby invite their Facebook friends to participate in the campaign.



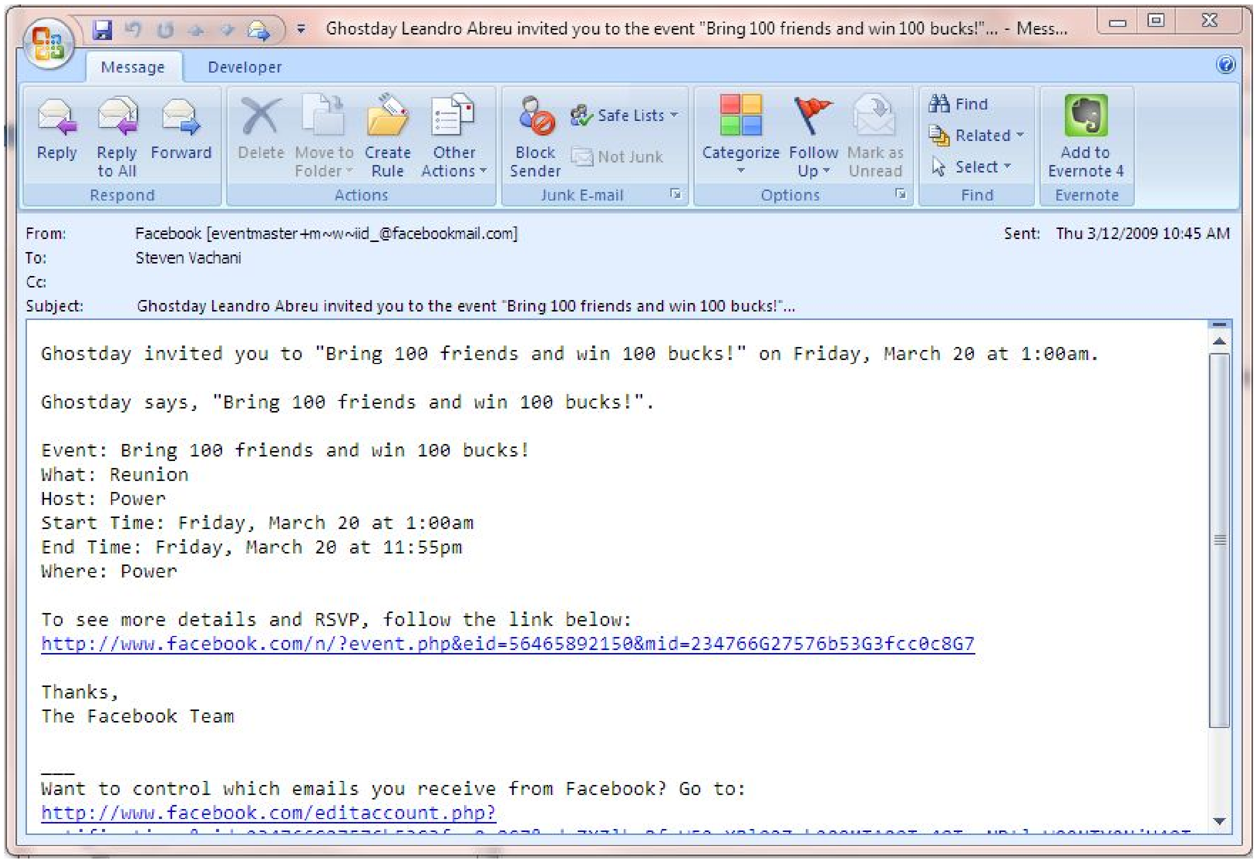
From: Facebook [wallmaster+m~w~iid_@facebookmail.com]
Sent: Sat 12/13/2008 7:41 PM
To: Steven Vachani
Cc:
Subject: Andrew Maciver wrote on your event's Wall...

```
Andrew wrote something on the Wall for the event "Bring 100 friends and win 100 bucks!".

To see what Andrew wrote, follow the link below:
http://www.facebook.com/n/?event.php&eid=44763165162#wall_posts

Thanks,
The Facebook Team


___
Want to control which emails you receive from Facebook? Go to:
http://www.facebook.com/editaccount.php?
notifications&md=ZXZlbnRfd2FsbDtlaWQ9NDQ3NjMxNjUxNjI7ZnJvbT01Nzg0MjUxMDc7dG89NjYwMDQwNTMx
```



From: Facebook [eventmaster+m~w~iid_@facebookmail.com]
Sent: Thu 3/12/2009 10:45 AM
To: Steven Vachani
Cc:
Subject: Ghostday Leandro Abreu invited you to the event "Bring 100 friends and win 100 bucks!"...

```
Ghostday invited you to "Bring 100 friends and win 100 bucks!" on Friday, March 20 at 1:00am.

Ghostday says, "Bring 100 friends and win 100 bucks!".

Event: Bring 100 friends and win 100 bucks!
What: Reunion
Host: Power
Start Time: Friday, March 20 at 1:00am
End Time: Friday, March 20 at 11:55pm
Where: Power

To see more details and RSVP, follow the link below:
http://www.facebook.com/n/?event.php&eid=56465892150&mid=234766G27576b53G3fcc0c8G7

Thanks,
The Facebook Team


___
Want to control which emails you receive from Facebook? Go to:
http://www.facebook.com/editaccount.php?
```

Figure 4: Screenshots of emails sent as a result of creating a Facebook Wall post and Facebook Event to invite friends to join Power 100 campaign

42. We also found another promotional message authored by the Defendants in the same resource file referenced above that was also used to invite Facebook friends to Power. *See* Exhibit E, at lines 147-149.

```
147  <data name="INVITEMESSAGE" xml:space="preserve">
148  <value>Hi ##friendname##,#BREAK#How would you
like all your friends in just one place?#BREAK#Login
to Power.com to discover all of its advantages and
enhance your Internet experience.</value>
149  </data>
```

43. This message includes a placeholder to insert a "`friendname`." In this case, we were able to find the Power.com software that would initiate sending these messages. The function that uses this "`INVITEMESSAGE`" string is named "`SendMessageInviteToPower`()." See Exhibit F, `PowerCallBack.aspx.cs`, found in the directory `SVN\power.com\Power.Com\Pub\Http`, at line 2623. The code excerpt below shows that the "`INVITEMESSAGE`" is used to form the body of the message to be sent as part of the invitation (see Exhibit F, at line 2681):

```
dataMessage.BodyMessage = Translate("INVITEMESSAGE")
.Replace("##name##", name)
.Replace("##friendname##", friendName)
```

44. This line of code retrieves the appropriate language translation for the "`INVITEMESSAGE`" (English, Spanish, or Portuguese) message, and then replaces the "friendname" placeholder in the text of the message that is sent with the actual friend's name in order to complete the content of the invitation to join Power.com. See Exhibit F, at line 2716, it calls the following function to send the message:

```
PowerMessageManager.SendMessage(dataMessage);
```

45. How the invitation is sent depends upon the network (e.g. Facebook) to which the invited friend belongs. The `PowerMessageManager.SendMessage()` method, which can be used to send invitations to users on Facebook, can be found starting at line 24 in the

`PowerMessageManager.cs` file found in the directory `SVN\power.com\Power.Message.Core`, which is attached hereto as Exhibit G. The related `SendMessage()` code is responsible for calling the `PowerMessageFactory.CreatePowerMessage()`, see Exhibit G, at line 42.

Further, a `CreatePowerMessage()` method that appears in the code uses the relevant network name (*e.g.* "Facebook") to determine how and where to send the electronic invitation. For the case where the network is Facebook, the following code would be executed (see Exhibit H, `PowerMessageFactory.cs`, found in the directory `SVN\power.com\Power.Message.Core`, at lines 45-50). This code shows Defendants would actually send an electronic message to someone from Facebook inviting them to join the Power.com website. The code to send the message uses a PowerScript which posts the message to the Facebook Wall of the friend to be invited. The code to retrieve and execute the PowerScript can be found in Exhibit I, `Write.cs`, found in the directory `SVN\power.com\Power.Message.Core\Engines`, at lines 87-94.

46. The code identifies the name of the PowerScript `PN_SEND_SCRAP_FACEBOOK` as that which was used for initiating the electronic invitations to join Power.com. The code was retrieved from a database where it was added by using the following SQL command (see Exhibit J, file `InsertMessageScript.sql`, found in directory `SVN\power.com\Power.Message.Core\Database`, at line 7). The `PN_SEND_SCRAP_FACEBOOK` script itself was retrieved from the PowerScript database (see Exhibit K, file `PN_SEND_SCRAP_FACEBOOK.xml`, at lines 1-23).

47. The PowerScript automatically posts the message content from the `INVITEMESSAGE` string to the Wall of a Facebook friend. Using these automatically generated messages, Defendants initiated electronic invitations for Facebook users to join the Power.com website.

C. POWER DATABASES

48. In addition to the code analysis, we also examined the related databases provided in an effort

to determine how many Facebook Event or Wall electronic mail messages were initiated by the PowerScript software. We determined that while certain of the databases were the ones of interest in which we would have expected to locate information about the numbers of electronic invitations that were sent by Power.com to Facebook, the databases produced by Defendants that should contain logs of the number of Events and Power.com invitations sent actually do not contain the information for the time period in question.

49. For instance, the Power.com database named Async is a log of PowerScript jobs run. The Async log would contain the information related to the number of electronic messages sent by the PowerScript software. The Async database found on the SQL 7 DVD only logged jobs from 2/19/2011 to 4/1/2011, and the Async database in SQL 6 DVD was corrupted. However, the disk that was provided that fixed the corrupted version only included logs from 08/03/2007 to 11/23/2008 – which does not cover the December 2008 period when the Facebook activity was seen. We understand that, according to information received from Mr. Timothy Fisher, Defendants stopped logging the PowerScript transactions into the database in November of 2008 as a result of migration of the company's servers to amazon.com as a host for the website. Whether true or not, the loss of information is prejudicial to Facebook, as only Defendants ever maintained such database logs.

50. In addition, we reviewed the content of the Power_Logger database in the expectation that it might include the information about the number of Facebook Events and Wall messages that the PowerScript software initiated. We did so because this database appears to include tables to log information about messages sent, including 10 MessageLog tables, a MessageLogHistory table, 10 Scraplog tables, and a ScraplogHistory table. Nonetheless, all of these tables were empty except for the ScrapLoghistory, which only had 141 entries from 12/6/2009 to 11/9/2010, all on the Orkut network.

51. Again, we understand that based on information received from Mr. Fisher, Defendants ceased daily operations sometime in April of 201l, at which time Defendants transferred all files onto a separate backup service. We further understand that, according to Mr. Fisher, the

Power_Logger database was supposedly too large to transfer, and therefore was removed. In our opinion, by deleting the Power_Logger database, Defendants effectively erased arguably the most relevant and useful information concerning the number of electronic mail messages that Defendants initiated through execution of their PowerScirpt software associated with the 100x100x100 campaign.

52. Because the information about Events and Wall messages sent to Facebook during December of 2008 was not included in the databases we received from Power, we were unable to determine precisely how many wall messages were posted and how many “Power 100” campaign Event notifications actually were sent to Facebook users.

53. However, we also examined the Power database of the Power website’s users, and we were able to determine that there were at least 39,137 Power.com users with Facebook accounts in the database. These database records include the stored email addresses used by the Power.com users in order to login on Facebook, and the stored passwords for their Facebook accounts.

D. DEFENDANTS’ EFFORTS TO CIRCUMVENT IP BLOCKS

54. We have found that the Power.com website utilized a pool of proxy servers to connect with social network sites, including Facebook, through different IP addresses. The Power software allowed the Defendants to configure a list of proxy servers for each social network site (see Exhibit L, file `HttpProxyConfig.cs`, found in the directory `SVN\powerinfra\Projectos\Power.PowerNetwork.Core\bll`, lines 107-141).

55. In one of the databases provided by Defendants, we were able to find an entry for the proxy server used to access Facebook. The IP address for the server was `174.129.224.81`, and a reverse directory lookup of this IP address identifies the host as `ec2-174-129-224-81.compute-1.amazonaws.com`. This IP address is associated with Amazon Web Services (see Exhibit M, file `AsyncSetup AsyncHttpProxy.csv`, extracted from the

database `SQL 7\AsyncSetup_full_bkp.bak`, database AsyncSetup, table: `AsyncHttpProxy`, row: 1). However, since the database only included a single IP address, and we understand that there were other IP addresses that Facebook attempted to block in December of 2008, it is clear this is not a complete list of the IP addresses that Defendants used to access Facebook. Additionally, the database did not include any history of which IP addresses were used for the critical time period of December of 2008 prior to when Defendants switched to Amazon Web Services. Based upon the proxy system software, it is clear that the Defendants could remove servers from service and replace servers both manually and automatically to circumvent IP blocks, such as those employed by Facebook.

56. The Defendants' software includes a command processing system to manage the server pool (see Exhibit N, `ServerManager.java` found in the directory `SVN\powerinfra\trunk\Java\powerproxy\com\powerscrap\proxy\manager`, lines 43-84). This command processing system is used to check status and make changes to any of the servers in the pool which may be blocked by a website such as Facebook, ("`BLOCK SERVER`") and to obtain a new server IP address when such a block is detected ("`GETNEXTIP`"). The commands can be issued either programmatically or manually. The command processing system effectively permitted Defendants to circumvent any attempts by websites like Facebook to block access by Defendants to those websites.

57. We also uncovered further evidence that Defendants implemented technology to circumvent Facebook's efforts to block the Power.com website by tracing the execution of the software used to create Facebook Events. Specifically, the latest delivery included the source code files that run the `CREATE_EVENT_FACEBOOK` script (see Exhibit O, `CreateCampaignEvent.cs` from directory `SVN\power.com\Power.Com.Core\Campaign100x100x100`, at lines 25-40. In this code, the PowerScript retrieved and executed the "`CREATE_EVENT_FACEBOOK`" script from one of Defendants' servers. Significantly, the IP address of the relevant server that

executes the "`CREATE_EVENT_FACEBOOK`" script is set by the Defendants' proxy server software. This functionality shows that the PowerScript software is intentionally monitored by the Power.com system to ensure that it is not being blocked by Facebook as a result of the software creating Facebook Events.

58. Additionally, we investigated certain routines in Defendants' source code to determine whether Defendants employed tools that allowed Defendants to circumvent technical barriers – such as blocks of IP addresses – that Facebook or other websites put in place to block Power's access to their own websites. Certain ones of these routines create a list of proxy servers, which are continuously monitored to determine, among other matters, if they are blocked by a website like Facebook. We have been able to identify connection-type methods in the source code that allow Defendants to use a proxy server to change the IP addresses used by the Power.com website that are visible to and are detected by third parties like Facebook. By tracing the execution of a PowerScript, we found that part of the process was to use `ConfigurationPowerProxy` to get a proxy server to use for connecting with Facebook. The code found shows how an array of proxy servers is created from a list provided by the proxy manager and then the server is selected randomly from that list. See Exhibit P, `ConfigurationPowerProxy.cs` found in the directory, `SVN\powerinfra\trunk\Projetos\src\configuration`, at lines 20-26.

59. The Defendants' source code includes routines we have identified that create a list of proxy servers, which are continuously monitored to determine, among other matters, if they are blocked by Facebook. The `updateServerListThread` shows the server list is updated on a regular interval stored in the `timeToUpdate` property (see Exhibit Q, `UpdateServerListManager.java`, found in the directory `SVN\powerinfra\trunk\Java\PowerInfra\powerproxy\com\powerscrap\proxy\manager`, at lines 107-118).

60. While the previous routine is used to update the server list on regular intervals, there are two other methods that are used to update the server list. The `definirServidor` method is

used to add a server to the list. The `removerServidor` method is used to remove a server from the list (see Exhibit R, `PowerProxy.java`, found in the directory `SVN\powerinfra\trunk\Java\PowerInfra\powerproxy\com\powerscrap\proxy`, at lines 82-108 and lines112-135, respectively). Finally the listen method at Exhibit R, lines 144-165 monitors each proxy server and calls the `removerServidor` routine if it detects a block of the Power.com website. The IP address of the Power.com website can then be replaced with another IP address from the Power Proxy Manager. In this way, Defendants ensure that they can circumvent deliberate blocks of its services by entities such as Facebook.

61. Based on the code examined it is clear that significant effort went into the design and development of the proxy system, and that one of the objectives of the system was to reconfigure IP connections if one of Defendants' proxy server's IP addresses used to connect to a website like Facebook was blocked. From our own understanding of the technology, we know that it is common for entities like Power.com to employ proxy pools to circumvent the blocking of IP addresses, especially when such entities are also employing scraping programs to obtain web content. As shown in our analysis above, the PowerScript scripts used by Power to create Facebook Events and write on Facebook friends' Walls are such web scraping programs. Moreover, Defendants' proxy pool infrastructure was designed to support these scraping activities.

62. Starting with the random selection of a proxy server to run each PowerScript, through the server list maintenance software described above, to the pool management command software and proxy server monitoring code also discussed above; Defendants' proxy system clearly was specifically designed to circumvent IP address blocking by entities such as Facebook.

E. DEFENDANTS HAVE DELETED IMPORTANT DATA

63. Since first getting access to some code on August 23, 2011, we have continually found

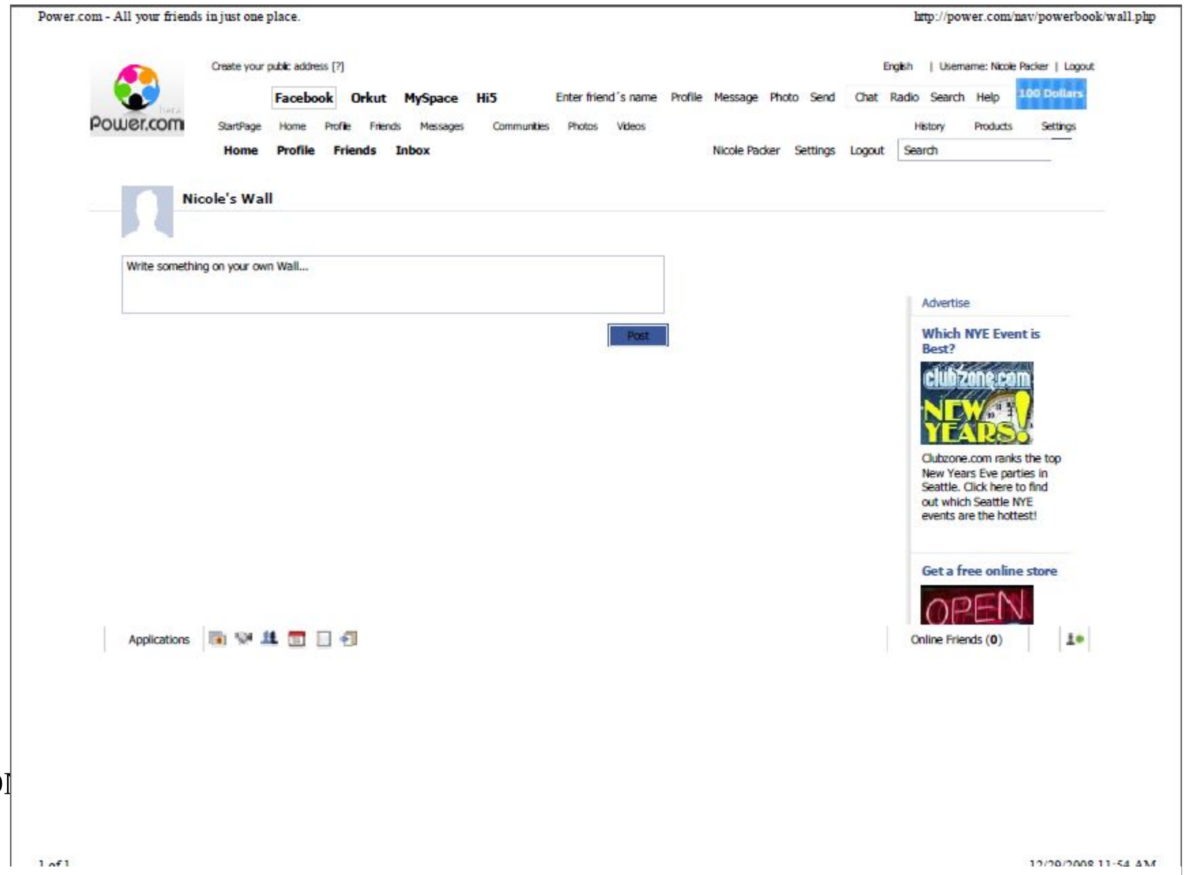
deficiencies in the scope of code produced by Defendants. For instance, as noted, despite repeated and diligent requests, we still have not received all of the Power database information associated with how many invitations were sent by Power. From correspondence from Mr. Fisher, we now believe this highly important information was deleted after this litigation was underway.

F. TECHNICAL ANALYSIS OF DECLARATION OF MR. VACHANI

64. Finally, we offer some observations about the obvious technical errors contained in statements by Defendant Steve Vachani in his December 12, 2011 Declaration. For instance, in paragraph 3 of Mr. Vachani's declaration, he states:

> Specifically, Power created a browser that allowed users to login and access all of their various social networking accounts at once. Users could update their photos, messages, music, and videos, and these updates would be portable across various social networking sites.

65. We believe Mr. Vachani's statements reflect his lack of technical acuity and programming skills to functionality understand the functionality of the PowerScript software. From our


Power.com - All your friends in just one place.
http://power.com/nav/powerbook/wall.php

Nicole's Wall

Write something on your own Wall...

Which NYE Event is Best?

Clubzone.com ranks the top New Years Eve parties in Seattle. Click here to find out which Seattle NYE events are the hottest!

Get a free online store

1 of 1
12/29/2008 11:54 AM

Create your public address [?]
English | Username: Nicole Packer | Logout
Facebook Orkut MySpace Hi5
Enter friend´s name Profile Message Photo Send
Chat Radio Search Help
100 Dollars
StartPage Home Profile Friends Messages Communities Photos Videos
History Products Settings
Home Profile Friends Inbox
Nicole Packer Settings Logout
Search
Post
Advertise
Applications
Online Friends (0)

Figure 5: Screenshot of Facebook Wall page included on a Power.com webpage

examination of the code, we did find that Defendants' software included a function that does resemble a browser – but only slightly. That software is contained in one of the many software directory trees and is named "`Navigator`." The software provided the user with the ability to open another web page and display it inside a Power.com web page. In Figure 5, a screenshot of the Power.com web page with a Facebook page embedded is shown (*see* FBPOWER00099). The screenshot, we believe reflects the functionality Mr. Vachani is referring to in his statement. However, while one function of this software was to allow the user to enter a message, the description of it as a "browser" does not capture the software's programmed crawling and scraping of websites. The Power software that provides this functionality is the "PowerScript" software, scripts, and infrastructure. As we have shown in our analysis above, PowerScripts were written to programmatically perform all the functions required to get and save photos, get and send messages, and get and save video, so as to "proxy" a website like Facebook. This functionality is directly related to a web scraper or webbot, and not a browser. That is so because the program or scripts perform the operation, rather than a user.

66. Also in paragraph 22 of his December 12, 2011 Declaration, Mr. Vachani states:

> Power did not access any nonpublic portion of Facebook's website. Power merely offered users a different and potentially superior browser through which they could access their Facebook accounts to copy, update, and/or port their own "User Content." And users did so by entering their own valid usernames and passwords, which Power never copied or stored for any purpose. Power did not obtain any software, data, or other content of value from Facebook. The only data accessed through Power's utilities were user's own "User Content," over which Facebook has disclaimed any ownership.

67. Again this statement is misleading because it represents that the user was controlling the access to public portions of Facebook's website. What actually occurred is that Defendants' software programmatically created Facebook Events. Unlike a browser, Defendants' software can programmatically create an Event, creates the list of friends to invite, and execute or send the Event invitations, without requiring any user interaction.

68. In addition, Mr. Vachani incorrectly states that Power never copied or stored Facebook usernames and passwords, whereas our examination of the "`power`" database found 39,137 Facebook login names and passwords stored there. An example of three network connections for one Power.com user is shown in Table 2. As can be seen from the table headings, this user belonged to three social networks: Facebook, LinkedIn, and Orkut. It also shows the username and password for each was stored within Defendants' databases. See power database, `dbo.account` table.

id	iduser	Nameaccountnetwork	username	idnetwork	password
17243056	977586	FACEBOOK	luiz.grecco@gmail.com	NULL	giF9l6JMQK8=
17134637	977586	LINKEDIN	luizg@sebraesp.com.br	1288243	q5HQ0dzE0bo=
906629	977586	ORKUT	luiz.grecco@gmail.com	1.35E+19	giF9l6JMQK8=

Table 2: List of three social network accounts, usernames and passwords stored in power database

69. In the same "`power`" database, we also found that friend lists were also stored. We found 225 records of Facebook friend lists that totaled 31,515 Facebook friends. An example of one of the 225 Facebook friend lists can be found in Exhibit S, friendlist2.xml from the power database, `dbo.FriendsAccount` table. This example friend list contains data for 253 Facebook friends and stores their Facebook IDs, Names and links to their Facebook profile photos.

70. Finally, in paragraph 11 of Mr. Vachani's December 12, 2011 declaration, he states:

> Power did not undertake any effort to circumvent that block, and did not provide users with any tools designed to circumvent it. Nevertheless, Facebook's IP block was ineffective because it blocked only one outdated IP address Power had used, and did not block other IPs that Power was using in the normal course of business.

71. From our analysis of the Power Proxy infrastructure, Defendants developed a flexible system for operating, managing, and maintaining a pool of proxy servers that could be assigned and removed from use easily. Because Defendants fundamentally relied on their software's ability to scrape information from other sites, the Power Proxy infrastructure limited the number of transactions coming from any one proxy server to reduce the likelihood of

detection by the websites being scraped. Defendants also put in place monitors to detect when errors occurred on one of the proxy servers, which included the ability to remove a failed proxy server from the list of servers. This type of proxy pool is not commonly used by websites, but is commonly used by web scraping services.

G. CONCLUSION

72. Based upon the review of Defendants' source code for various code projects named PowerScript, PowerNavigtor, PowerProxy, and spider, as well as other documentation produced to date, we have concluded the following:

(a) Defendants developed proprietary software named PowerScript and spider in order to crawl various social network websites, including particularly www.facebook .com , to extract or "scrape" website user information such as Facebook photo images, wall content, friends' lists, and the like, and to then reformat that user information on Defendants' own website, www.power.com, in order to "proxy" Facebook and permit Defendants' own website users to log into Facebook through Defendants' own Graphical User Interface, rather than through Facebook's interface.

(b) Defendants designed their proprietary PowerScript and spider software to automatically post on the Facebook website new Events soliciting Facebook users to join Power.com as part of what Defendants called the "Power 100" or "100x100x100" Campaign. Defendants likewise designed their software to automatically post Power Invitations on Facebook users' Walls soliciting them to join Power.com.

(c) Based upon available information from Defendants' databases, at least 39,137 users of the Power website also had Facebook accounts. Because of missing information from those databases that is solely in the control of Defendants, we were unable to quantify exactly how many Facebook Event or wall posting transactions took place between the Power website and Facebook in which Facebook users were solicited to join Power.com. We are able to state that both kinds of solicitations did occur, however, and were initiated by

Defendants' proprietary software.

(d) In addition to the electronic mail communications that Defendants' software automatically posted on the Facebook websites when it created Facebook Events and when it posted Facebook wall messages, the same proprietary software that Defendants used to automatically create Event notifications and post Facebook Wall messages also would initiate automated "spam" email messages being sent on Defendants' behalf to Facebook.

73. It is our understanding that discovery in this case is ongoing. Accordingly, we reserve the right to supplement or amend our opinions in light of any additional evidence, testimony, or information that may be provided to us after the date of this report. We also reserve the right to supplement or amend our opinions in response to any expert reports served by any other party in the lawsuit.

Dated: 19-Dec-2011

[illegible]

Robert Zeidman

Lawrence Melling

Defendants' proprietary software.

(d) In addition to the electronic mail communications that Defendants' software automatically posted on the Facebook websites when it created Facebook Events and when it posted Facebook wall messages, the same proprietary software that Defendants used to automatically create Event notifications and post Facebook Wall messages also would initiate automated "spam" email messages being sent on Defendants' behalf to Facebook.

73. It is our understanding that discovery in this case is ongoing. Accordingly, we reserve the right to supplement or amend our opinions in light of any additional evidence, testimony, or information that may be provided to us after the date of this report. We also reserve the right to supplement or amend our opinions in response to any expert reports served by any other party in the lawsuit.

Dated: 19-Dec-2011

Robert Zeidman

[illegible]

Lawrence Melling

Exhibit A: Resume of Robert Zeidman

Exhibit B: Larry Melling Resume

Exhibit C: Expert Report Source Code Inspection Log 2011-12-19

Exhibit D: CREATE_EVENT_FACEBOOK.xml

Exhibit E: PowerCallBack.aspx.en.resx

Exhibit F: PowerCallBack.aspx.cs

Exhibit G: PowerMessageManager.cs

Exhibit H: PowerMessageFactory.cs

Exhibit I: Write.cs

Exhibit J: InsertMessageScript.sql

Exhibit K: PN_SEND_SCRAP_FACEBOOK.xml

Exhibit L: HttpProxyConfig.cs

Exhibit M: AsyncSetup AsyncHttpProxy.csv

Exhibit N: ServerManager.java

Exhibit O: CreateCampaignEvent.cs

Exhibit P: ConfigurationPowerProxy.cs

Exhibit Q: UpdateServerListManager.java

Exhibit R: PowerProxy.java

Exhibit S: friendList2.xml